AD-A210 024

FJSRL-TM-89-0003

FRANK J. SEILER RESEARCH LABORATORY

A Basic Program for Desktop Computers to Control Battery Cycling Experiments

DTIC
ELECTE
JUL 12 1989
S D
CD

R. Larry Vaughn

APPROVED FOR PUBLIC RELEASE
DISTRIBUTION UNLIMITED

June 1989

AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

89 7 11 050

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Distribution Unlimited Approved for public release
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) FJSRL-TM-89-0003	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Frank J. Seiler Research Lab 6b. OFFICE SYMBOL (If applicable) FJSRL	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) USAF Academy Colorado 80840-6528	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Air Force Office of Scientific Research 8b. OFFICE SYMBOL (If applicable) AFOSR	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) Bldg 410 Bolling AFB DC 20332	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 61102F; PROJECT NO. 2302; TASK NO. F2; WORK UNIT ACCESSION NO. 10

11. TITLE (Include Security Classification)
A Basic Program for Desktop Computers to Control Battery Cycling Experiments

12. PERSONAL AUTHOR(S)
R. Larry Vaughn

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Technical Memorandum	FROM Jun 88 TO Jun 89	89, June	28

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Computer, ~~Cycling~~
09	02		Program,
10	03		Battery, electrical [illegible]

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

BATTERY.BAS is a BASIC program that automatically executes battery cycling experiments, takes voltage–current data, monitors open circuit voltage, and some combinations of these tests. Since the program is written in interpretive BASIC, the user has the flexibilty to modify some experimental parameters while the experiment is in progress.

Data are saved in various sequential ASCII files that can be accessed by any program that can read ASCII files. SlideWrite Plus™ (Advanced Graphics Software, Inc.), for example, is used for graphical presentation of data. Some utility programs are included to manipulate data collected by the program. Typical manipulations are separation of individual cycles from multiple cycle data and conversion of voltage–time data to voltage–charge data.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	Unclassified
22a. NAME OF RESPONSIBLE INDIVIDUAL R. Larry Vaughn	22b. TELEPHONE (Include Area Code) (719)472-2655
	22c. OFFICE SYMBOL FJSRL/NC

DD Form 1473, JUN 86 *Previous editions are obsolete.*

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

A BASIC PROGRAM FOR DESKTOP COMPUTERS TO CONTROL BATTERY CYCLING EXPERIMENTS

R. Larry Vaughn
Frank J. Seiler Research Laboratory
USAF Academy, CO 80840–6528

June 1989

FJSRL–TM–89–0003

DTIC COPY INSPECTED

Accesion For
NTIS CRA&I
DTIC TAB
Unannounced
Justification
By
Distribution /
Availability Codes
Dist | Avail and / or Special
A-1

TABLE OF CONTENTS

INTRODUCTION

BATTERY.BAS is a BASIC program that automatically executes battery cycling experiments, takes voltage–current data, monitors open circuit voltage, and some combinations of these tests. Since the program is written in interpretive BASIC, the user has the flexibilty to modify some experimental parameters while the experiment is in progress.

Data are saved in various sequential ASCII files that can be accessed by any program that can read ASCII files. SlideWrite Plus™ (Advanced Graphics Software, Inc.), for example, is used for graphical presentation of data. Some utility programs are included to manipulate data collected by the program. Typical manipulations are separation of individual cycles from multiple cycle data and conversion of voltage–time data to voltage–charge data.

HARDWARE DESCRIPTION

A PAR 173 Potentiostat/Galvanostat equipped with a PAR 276 plugin module (Princeton Applied Research) provides a constant current source. The PAR 276 has an IEEE port that allows a computer to control the potentiostat. Voltage is monitored by an HP3455A digital voltmeter (Hewlett Packard) that is also equiped with an IEEE port. A Zenith 248 controls the experiment and saves the data to a hard disk. A National Instruments IEEE–488 Instrumentation Interface (National Instruments) and associated software are installed in the computer. The software provided with the National Instruments interface includes the drivers for addressing the IEEE buss and the necessary lines that must be included in the user's BASIC program. The computer, DVM, and PAR 173 were plugged in to an Inmac Model 1000 Personal Uninterruptable Power Supply to protect against experiment shutdowns during short power outages.

The battery (or cell) is connected to the potentiostat with the green lead to the cathode (positive electrode) and the red lead to the anode (negative electrode). The black lead is not used. The electrometer probe is also not used and voltage is monitored by the voltmeter instead. The front panel controls of the PAR 173 are set to zero with the A Channel polarity set negative and the B Channel polarity set positive. Before starting an experiment, the cell switch on the PAR 173 must be in the EXT CELL position.

The program can be modified for use with a PAR 273. Voltage can be monitored directly with the PAR 273, thus eliminating the need for the digital voltmeter. All references in BATTERY.BAS that address the DVM must be eliminated or changed to read potential from the PAR 273. Cell connections will be slightly different, since all three leads from the potentiostat must be connected to the test cell.

PROGRAM INSTALLATION AND DESCRIPTION

The National Instruments software should be installed according to the instructions provided in the user's manual. The user should insure that the CONFIG.SYS file contains the line

DEVICE=GPIB.COM

and that the file GPIB.COM is in the root directory. The file BIB.M should be in the same subdirectory as BATTERY.BAS. National Instruments provides a setup program, IBCONF, for the user to easily establish the device addresses of the potentiostat and voltmeter. The program DECL.BAS supplied by National Instruments includes the drivers and declarations for the IEEE standard interface and are included in the users progrm in lines 1–99. The documentation gives instructions to determine the memory requirements in line 1 of the program.

Appendix A contains a complete listing of BATTERY.BAS, including lines 1–99 supplied by National Instruments. Only line 1–6 are required for the program as written. The program is divided into the following segments:

The main program:

- Lines 1 – 99: DECL.BAS. Not all lines are needed, but are included here for completeness. Only lines 1 – 6 must be included.
- Lines 100 – 200: dimension arrays and obtain the user's choice of experiment(s).
- Lines 210 – 270: call the subroutines that get the experimental parameters and run the experiment(s) chosen by the user.

Subroutines to get the users input for experimental parameters:

- Lines 700 – 795: battery cycling parameters.
- Lines 800 – 870: voltage–current parameters.
- Lines 900 – 950: open circuit voltage monitor parameters.

Subroutines that execute the experiments:

- Lines 1000 – 1380 : battery cycling.
- Lines 2000 – 2135 : voltage–current data
- Lines 3000 – 3220 : open circuit monitor.

Subroutines for handling devices, errors, and output:

- Lines 7000 – 7090 : initialize potentiostat and voltmeter
- Lines 8000 – 8040 : clear potentiostat and voltmeter
- Lines 9000 – 9070 : handle device error
- Lines 9100 – 9140 : print warning message
- Line 9200 : prints input screen title

The program produces a number of data files depending on which experiments are executed. For battery cycling TESTNAME$.HDR (TESTNAME$ is the name the user selects to identify the experiment) contains basic information about the cycling experiment. Data saved in TESTNAME$.HDR are the testname, the values of the charging and discharging current and cut off voltages, and maximum number of cycles in the test. For each cycle open circuit voltage and number of data points for that cycle are appended to the file. The file TESTNAME$.HIS contains the voltage–time (in minutes) data for every cycle in the experiment. The file TESTNAME$.VI, produced in the voltage–currrent experiment, contains the voltage current data as the current, open circuit voltage, load voltage, and the difference between the two voltages for each current. The file TESTNAME$.OCV, created in the open circuit monitor routine, contains the open circuit potential and the elapsed time in hours.

RUNNING THE PROGRAM

The subdirectory containing BATTERY.BAS should be selected as the default directory. Then, typing

BASICA BATTERY<RETURN>

at the DOS prompt immediately runs the program.

The program presents the user with the experimental choices available to which the user must respond with the number for the desired experiment followed by **<RETURN>**. Once the user makes a choice successive input screens prompt the user for the parameters required by the experiment(s) selected. The program initializes the potentiostat and voltmeter at the beginning of the experiment.

ENTERING EXPERIMENTAL PARAMETERS

The program will prompt the user for all the experimental paramters required by the experiments to be run. Pressing **<RETURN>** at a prompt will accept the default value shown in brackets by the prompting message.

Parameters for cycling experiment

Testname: this should be a decriptive name for the experiment. Since this name will be used as the file name for the data it must conform to the file–naming rules of DOS. No extension should be used here, since the program supplies the extension for the experiment being run. Value is stored in variable TESTNAME$.

Discharge Current: the current in milliamps to be used in discharging the cell. Value is stored in variable DCUR.

Discharge Cut–off Voltage: the voltage at which discharge will stop. Value is stored in variable DCOV.

Charge Current: the current in milliamps to be used in charging the cell. Value is stored in variable CCUR.

Charge Cut–off Voltage: the voltage at which charging ends. Value is stored in variable CCOV.

Maximum Time Between Points: the maximum time interval in minutes between data points. If no other event has caused a data point to be saved, the expiration of this time interval will cause a data point to be saved. Default is 5 minutes. Value is stored in variable MAXTIME%

Maximum Volts Between Points: a voltage change that will cause a data point to be taken if no other event has caused a data point to be saved. Default is 0.10 volt. Value is stored in variable MAXVOLT

Maximum Number of Cycles: the total number of complete charge/discharge cycles to be completed. Default is 50. Value is stored in variable MAXCYCLE%

Delay Between Cycles: number of seconds the program waits from the end of one cycle to the beginning of another cycle. This is included to allow the cell to reach a stable open circuit voltage before starting another cycle. Default is 300 seconds. Value is stored in variable DELAY%

Start first cycle with [C]harge or [D]ischarge: gives the option of starting a cycling experiment with a charge or a discharge. Default is "D" for discharge. Entering either lowercase or uppercase is acceptable. Value is stored in variable FIRSTCYCLE$.

> NOTE: A single charge or a single discharge can be accomplished with the proper selection of values for the parameters DCOV, CCOV, DCUR, and CCUR. To charge a cell without a discharge, make DCUR small (0.0001 for example) and DCOV large (equal to or greater than CCOV for example). To discharge a cell without charging make CCUR small (0.0001 for example) and CCOV small (equal to or less than DCOV). Setting either DCUR or CCUR to zero may result in an error condition in the PAR 276.

Parameters for voltage–current experiment

Testname: same as for cycling experiment. If it has already been entered, it will show between brackets and pressing <RETURN> accepts the value shown. Normally this name will be the same for all experiments. The program adds extentions to the filename depending on which experiment(s) is/are run.

Initial current: the current in milliamps to start the experiment. Value is stored in variable STARTI.

Current increment: the amount to be added in each current increment. Value is stored in variable CURINCR.

Number of points: the number of data points that will be taken. Fifteen is the maximum allowed. Value is stored in variable PT2%.

Time on load: the time in seconds that the current will be applied to the test cell at which point the voltage will be recorded. Default is 3. Value is stored in variable LOADTIME%

Recovery time: the time in seconds that the cell will be on open circuit before the next current is applied. Default is 30. Value is stored in variable RECOVER%.

Parameters for open circuit monitor

Testname: same as for previous experiments.

Termination time for run: time in hours that the open circuit voltage will be monitored unless some other event causes earlier termination of the experiment. Default is 12. Value is stored in variable TERMTIME%.

Termination voltage for run: the voltage at which the experiment will end unless some other event causes earlier termination of the experiment. Default is 0. Value is stored in varialbe TERMVOLT.

Maximum time between points: the time in minutes at which a data point will be taken if no other event has caused a point to be taken. Default is 30. Value is stored in variable MAXTIME3%.

Maximum voltage between points: the voltage change at which a data point will be taken if no other event has caused a point to be taken. Default is 0.5. Value is stored in varialbe MAXVOLT3.

Once the experimental parameters have been entered, the program pauses with the following prompt to remind the user to insure the test cell is connected and the PAR 173 is set to EXT CELL before continuing.

> Insure PAR 173 on EXT CELL and leads are connected to cell
> Press <RETURN> to start experiment

Pressing the return key at this point causes the program to immediately start the experiment by successively running the experimental subroutines corresponding to the user's choice at the beginning.

> NOTE. While this message is printed for the open circuit monitor experiment, it is not necessary for the PAR 173 be set to EXT CELL to monitor the open circuit voltage. However, the PAR 173 power switch MUST be on.

Appendix B contains a detailed explanation of the experimental subroutines.

CHANGING PARAMETERS WHILE EXPERIMENT IS RUNNING

Since the program is written in interpretive BASIC it has the flexibility to allow the user to change many experimental parameters while an experiment is in progress. This is accomplished by pressing **<CTRL> <BREAK>**. The BASIC prompt will appear. Change the parameter desired using normal BASIC assignment syntax. For example, to change the termination time for an Open Circuit Monitor experiment enter the following

termtime% = ## <RETURN>

where ## is the new value for the termination time. Resume the experiment by typing

cont <RETURN>

at the Ok prompt.

Parameters commonly modified are DCOV, CCOV, CYCLE%, and TERMTIME%. Most parameters entered in response to the parameter input prompts previously described can be changed. Parameters that should not be changed while an experiment is running are DCUR and CCUR.

UTILITY PROGRAMS TO MANIPULATE BATTERY CYCLING DATA AND SAMPLES OF OUTPUT

Several short utility programs are used to manipulate data files produced by the BATTERY.BAS. These programs are used to convert the product file from the form saved by BATTERY.BAS to another form for analysis or presentation. Program listings for the utility programs are in Appendix C. Files produced by these utilities have filenames TESTNAME$ plus an extension. TESTNAME$.HIS is the data file produced by BATTERY.BAS for cycling experiments and contains every cycle successively. Figure 1 shows a SlideWrite chart made from data contained in a TESTNAME$.HIS data file. These utility programs are run as any BASIC program. Each program prompts the user to enter the testname of the data that is to be converted. If no extension is given with the testname the program assumes a "HIS" extension. The program SEPARATE.BAS operates only on "HIS" files so no extension should be given at the TESTNAME prompt. The other programs can accept other extensions. This allows these programs to operate on separated cycles resulting from the use of SEPARATE.BAS. In all cases, however, the entry for the testname should include the path for the data file if it is not in the current directory. For example, if the data file is on a floppy in the A drive while the utility program is on the C drive, the response to the Testname prompt should be:

a:testname <RETURN>

> **WARNING** Since the ?CONVERT programs save data tc files with fixed extensions, running one of these programs will overwrite the file that was made in any previous conversion using the same program with the same TESTNAME$ filename.

SEPARATE.BAS – separates multiple cycles into separate files for each cycle and puts each cycle into its own data file TESTNAME.## where ## is the cycle number. Figure 2 shows a SlideWrite chart for a single cycle extracted from the data depicted in Figure 1.

VCONVERT.BAS – converts normal voltage–time curve to loop at the end of charge or discharge by reversing the time scale. The new data is saved in file TESTNAME$.DTV. Figure 3 shows the same data as in Figure 1 after conversion by VCONVERT.BAS.

QCONVERT.BAS – converts voltage–time data to voltage–charge data and stores the new data in TESTNAME$.DTQ. Figure 4 shows the data from from Figure 1 converted to show charge on the x–axis.

TCONVERT.BAS – converts time scale from minutes to hours and saves the new data in TESTNAME$.HRS.

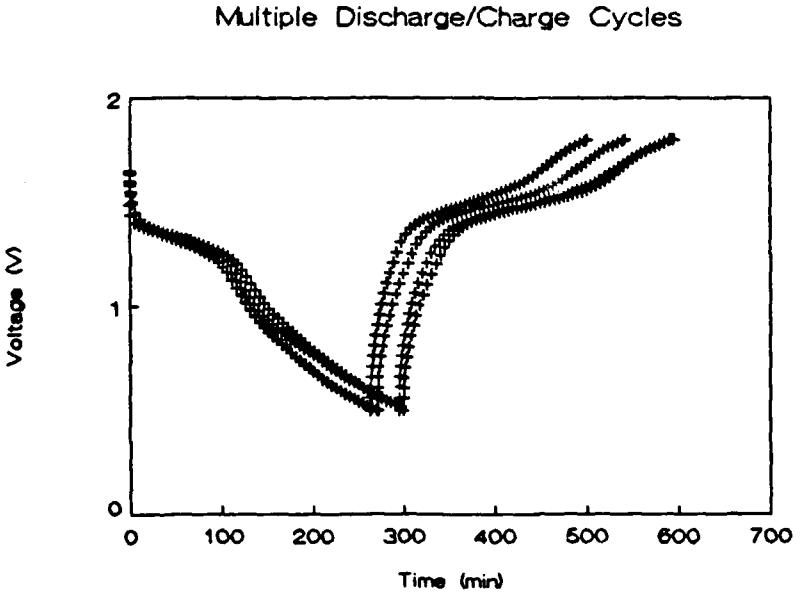

FIGURE 1. Graphical presenatation of multiple cycles as acquired by BATTERY.BAS.

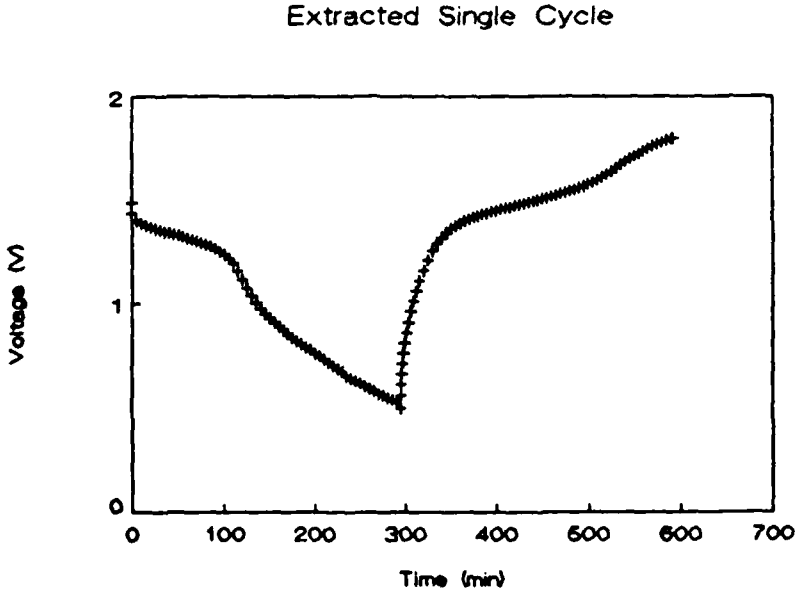

FIGURE 2. Single cycle extracted from the data depicted in Figure 1 by running SEPARATE.BAS.

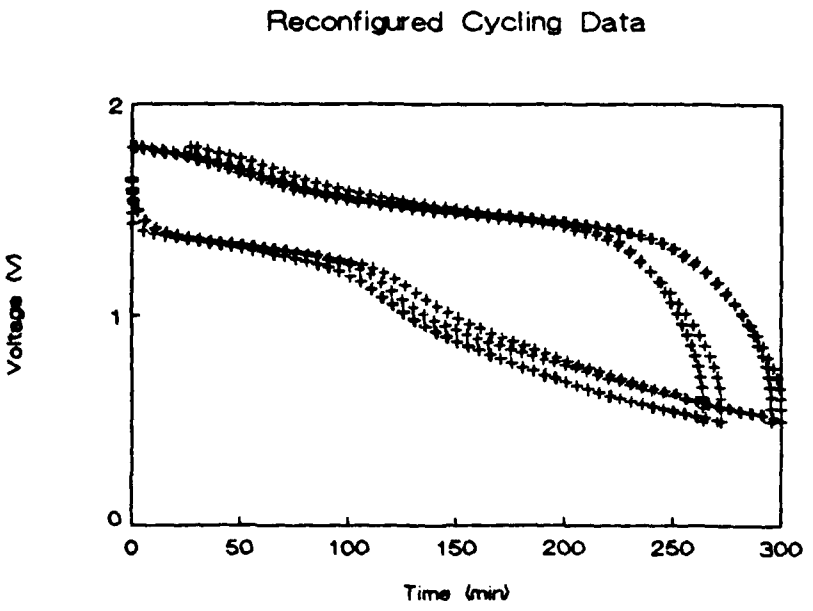

FIGURE 3. Data from Figure 1 reconfigured by running VCONVERT.BAS.

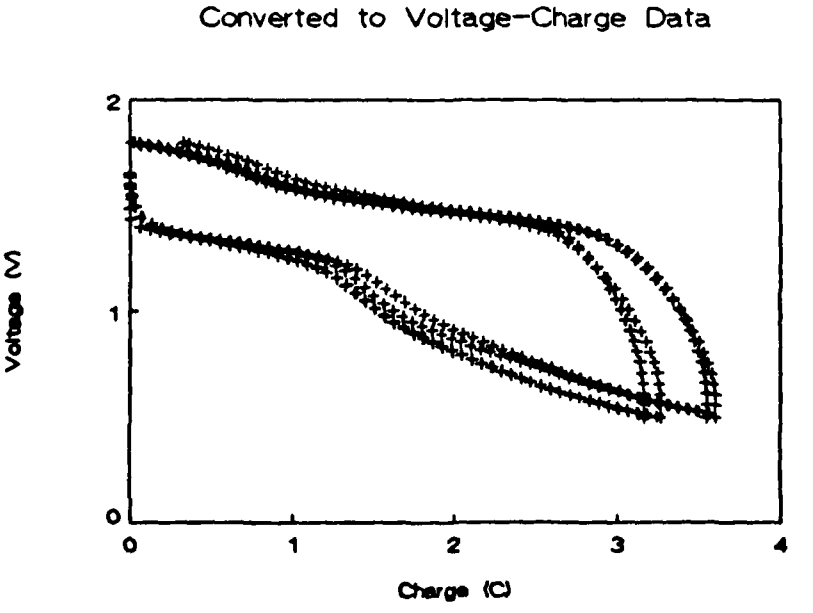

FIGURE 4. Data from Figure 1 that has been converted to Voltage-Charge format by QCONVERT.BAS.

APPENDIX A
BATTERY.BAS PROGRAM LISTING

```
1     CLEAR ,59100!        ' BASIC Declarations
2     IBINIT1 = 59100!
3     IBINIT2 = IBINIT1 + 3   ' Lines 1 through 6 MUST be included in your program.
4     BLOAD "bib.m",IBINIT1
5                                                                          CALL
IBINIT1(IBFIND,IBTRG,IBCLR,IBPCT,IBSIC,IBLOC,IBPPC,IBBNA,IBONL,IBRSC,I
BSRE,IBRSV,IBPAD,IBSAD,IBIST,IBDMA,IBEOS,IBTMO,IBEOT,IBRDF,IBWRTF)
6                                                                          CALL
IBINIT2(IBGTS,IBCAC,IBWAIT,IBPOKE,IBWRT,IBWRTA,IBCMD,IBCMDA,IBRD,I
BRDA,IBSTOP,IBRPP,IBRSP,IBDIAG,IBXTRC,IBRDI,IBWRTI,IBRDIA,IBWRTIA,I
BSTA%,IBERR%,IBCNT%)
7     REM Optionally include the following declarations in your program.
8     REM They provide appropriate mnemonics by which
9     REM to reference commonly used values. Some mnemonics (GET%, ERR%,
10    REM END%, ATN%) are preceded by "B" in order to distinguish them from
11    REM BASIC keywords.
12    REM
13    REM GPIB Commands
14    UNL% = &H3F     ' GPIB unlisten command
15    UNT% = &H5F     ' GPIB untalk command
16    GTL% = &H1      ' GPIB go to local
17    SDC% = &H4      ' GPIB selected device clear
18    PPC% = &H5      ' GPIB parallel poll configure
19    BGET% = &H8     ' GPIB group execute trigger
20    TCT% = &H9      ' GPIB take control
21    LLO% = &H11     ' GPIB local lock out
22    DCL% = &H14     ' GPIB device clear
23    PPU% = &H15     ' GPIB ppoll unconfigure
24    SPE% = &H18     ' GPIB serial poll enable
25    SPD% = &H19     ' GPIB serial poll disable
26    PPE% = &H60     ' GPIB parallel poll enable
27    PPD% = &H70     ' GPIB parallel poll disable
28    REM
29    REM GPIB status bit vector
30    REM global variable IBSTA% and wait mask
31    BERR% = &H8000  ' Error detected
32    TIMO% = &H4000  ' Timeout
33    BEND% = &H2000  ' EOI or EOS detected
34    SRQI% = &H1000  ' SRQ detected by CIC
35    RQS% = &H800    ' Device needs service
36    CMPL% = &H100   ' I/O completed
37    LOK% = &H80     ' Local lockout state
38    REM% = &H40     ' Remote state
39    CIC% = &H20     ' Controller-In-Charge
40    BATN% = &H10    ' Attention asserted
41    TACS% = &H8     ' Talker active
42    LACS% = &H4     ' Listener active
```

```
43    DTAS% = &H2     ' Device trigger state
44    DCAS% = &H1     ' Device clear state
45    REM
46    REM Error messages returned in global variable IBERR%
47    EDVR% =  0      ' DOS error
48    ECIC% =  1      ' Function requires board to be CIC
49    ENOL% =  2      ' Write function detected no Listeners
50    EADR% =  3      ' Interface board not addressed correctly
51    EARG% =  4      ' Invalid argument to function call
52    ESAC% =  5      ' Function requires board to be SAC
53    EABO% =  6      ' I/O operation aborted
54    ENEB% =  7      ' Non-existent interface board
55    EOIP% = 10      ' I/O operation started before previous operation completed
56    ECAP% = 11      ' No capability for operation
57    EFSO% = 12      ' File system operation error
58    EBUS% = 14      ' Command error during device call
59    ESTB% = 15      ' Serial poll status byte lost
60    ESRQ% = 16      ' SRQ remains asserted
61    REM
62    REM EOS mode bits
63    BIN% = &H1000   ' Eight bit compare
64    XEOS% = &H800   ' Send EOI with EOS byte
65    REOS% = &H400   ' Terminate read on EOS
66    REM
67    REM Timeout values and meanings
68    TNONE% =  0     ' Infinite timeout (disabled)
69    T10US% =  1     ' Timeout of 10 us (ideal)
70    T30US% =  2     ' Timeout of 30 us (ideal)
71    T100US% =  3    ' Timeout of 100 us (ideal)
72    T300US% =  4    ' Timeout of 300 us (ideal)
73    T1MS% =  5      ' Timeout of 1 ms (ideal)
74    T3MS% =  6      ' Timeout of 3 ms (ideal)
75    T10MS% =  7     ' Timeout of 10 ms (ideal)
76    T30MS% =  8     ' Timeout of 30 ms (ideal)
77    T100MS% =  9    ' Timeout of 100 ms (ideal)
78    T300MS% = 10    ' Timeout of 300 ms (ideal)
79    T1S% =  11      ' Timeout of 1 s (ideal)
80    T3S% =  12      ' Timeout of 3 s (ideal)
81    T10S% =  13     ' Timeout of 10 s (ideal)
82    T30S% =  14     ' Timeout of 30 s (ideal)
83    T100S% =  15    ' Timeout of 100 s (ideal)
84    T300S% =  16    ' Timeout of 300 s (ideal)
85    T1000S% = 17    ' Timeout of 1000 s (maximum)
86    REM
87    REM Miscellaneous
88    S% = &H8        ' Parallel Poll sense bit
89    LF% = &HA       ' Line feed character
90    REM
91    REM Application program variables passed to
92    REM GPIB functions
93    REM
```

```
94     CMD$ = SPACE$(10)       ' command buffer
95     RD$ = SPACE$(255)       ' read data buffer
96     WRT$ = SPACE$(255)      ' write data buffer
97     BNAME$ = SPACE$(7)      ' board name buffer
98     BDNAME$ = SPACE$(7)     ' board or device name buffer
99     FLNAME$ = SPACE$(50)    ' file name buffer
100 KEY OFF
105 DIM ETIME(2000),EVOLT(2000),CUR(15), OCV(15), LOADV(15)
107 CELLON$ = "CELL 1":  CELLOFF$= "CELL 0"
110 CLS: LOCATE 1,30: PRINT "BATTERY TEST PROGRAM"
120 LOCATE 4,34: PRINT "TEST CHOICES"
130 LOCATE 6,24: PRINT "(1)  Battery Cycling
140 LOCATE 7,24: PRINT "(2)  Voltage–Current Data
150 LOCATE 8,24: PRINT "(3)  Monitor Open Circuit Voltage
155 LOCATE 9,24: PRINT "(4)  (1) THEN (2)"
160 LOCATE 10,24: PRINT "(5)  (1) THEN (3)"
165 LOCATE 11,24: PRINT "(6)  (1) THEN (2) THEN (3)"
170 LOCATE 12,24: PRINT "(7)  Exit to DOS"
175 LOCATE 14,24:PRINT "Enter number of choice:";
180 INPUT Z%: IF Z% > 7 THEN BEEP:GOTO 180
190 IF Z% = 7 THEN GOTO 600
200 GOSUB 7000                  ' initialize PAR and DVM
209 ' ******** parameter input screens ********************
210 IF Z% = 1 OR Z% > 3 THEN GOSUB 700
220 IF Z% = 2 OR Z% = 4 OR Z% = 6 THEN GOSUB 800
230 IF Z% = 3 OR Z% = 5 OR Z% = 6 THEN GOSUB 900
240 GOSUB 9100          'print warning message
249 ' ******** run experiments ******************************
250 IF Z% = 1 OR Z% > 3 THEN GOSUB 1000
260 IF Z% = 2 OR Z% = 4 OR Z% = 6 THEN GOSUB 2000
270 IF Z% = 3 OR Z% = 5 OR Z% = 6 THEN GOSUB 3000
280 GOSUB 8000          'clear PAR and DVM
290 GOTO 110            'return to menu
300 END
599 ' ———————— RETURN TO DOS ————————————————————————
600 CLS: SYSTEM         ' return to DOS
700 PROGRAM$="BATTERY CYCLING — CONSTANT CURRENT"
701 ' ************ enter experimental parameters **********************
702     MAXTIME%=5:     MAXVOLT=.1:     MAXCYCLE%=50:     DELAY%=300:
FIRSTCYCLE$ = "D"
710 GOSUB 9200
715 LOCATE 3,1: PRINT "Enter experimental parameters – "
720 LOCATE 5,1: INPUT "TestName :";TESTNAME$
730 LOCATE 7,1: INPUT "Discharge Current (mA): ";DCUR
735 LOCATE 7,40:INPUT "Discharge Cut–off Voltage: ";DCOV
740 LOCATE 8,1: INPUT "Charge Current (mA)  : ";CCUR
745 LOCATE 8,40:INPUT "Charge Cut–off Voltage  : ";CCOV
750 LOCATE 10,1: PRINT "Maximum Time Between Points (min) [";MAXTIME%;"] : ";
755 INPUT Z$: IF Z$ <> "" THEN MAXTIME%=VAL(Z$)
760 LOCATE 11,1: PRINT "Maximum Volts Between Points [";MAXVOLT;"] :";
765 INPUT Z$: IF Z$ <> "" THEN MAXVOLT = VAL(Z$)
```

```
770 LOCATE 12,1: PRINT "Maximum Number of Cycles [";MAXCYCLE%;"] :";
775 INPUT Z$: IF Z$ <> "" THEN MAXCYCLE%=VAL(Z$)
780 LOCATE 13,1: PRINT "Delay between cycles (seconds) [";DELAY%;"]: ";
785 INPUT Z$: IF Z$ <> "" THEN DELAY%=VAL(Z$)
787 LOCATE 15,1: PRINT "Start first cycle with [C]harge or [D]ischarge
[";FIRSTCYCLE$;"]: ";
789 INPUT Z$: IF Z$ <> "" THEN FIRSTCYCLE$ = Z$
791 IF INSTR("CcDd",FIRSTCYCLE$) = 0 THEN BEEP: GOTO 787
793 IF INSTR("Cc",FIRSTCYCLE$) = 0 THEN FIRSTCYCLE$="D" ELSE
FIRSTCYCLE$="C"
795 RETURN
799 ' ********** V-I parameters **********
800 PROGRAM$="BATTERY CURRENT - VOLTAGE DATA"
805 LOADTIME% = 3 : RECOVER% = 30 'DEFAULT VALUES
810 ' ********** enter experimental parameters **********
815 GOSUB 9200
820 LOCATE 3,1: PRINT "Enter experimental parameters - "
825 LOCATE 5,1: PRINT "TestName [";TESTNAME$;"] :";TESTNAME$;
827 INPUT Z$: IF Z$ <> "" THEN TESTNAME$ = Z$
830 LOCATE 7,1: INPUT "Initial Current (mA) : ";STARTI
835 LOCATE 8,1: INPUT "Current Increment (mA): ";CURINCR
840 LOCATE 9,1: INPUT "Number of Points (15 maximum) : ";PT2%
845 LOCATE 10,1:PRINT "Time on Load (sec) [";LOADTIME%;"] :";
850 INPUT Z$: IF Z$ <> "" THEN LOADTIME% = VAL(Z$)
855 LOCATE 11,1: PRINT "Recovery Time (sec) [";RECOVER%;"] :";
860 INPUT Z$: IF Z$ <> "" THEN RECOVER%=VAL(Z$)
870 RETURN
900 PROGRAM$="BATTERY OPEN CIRCUIT VOLTAGE"
905 TERMTIME% = 12: TERMVOLT = 0: MAXTIME3%=30: MAXVOLT3 = .5
910 ' ********** experimental parameters **********
915 GOSUB 9200
920 LOCATE 3,1: PRINT "Enter experimental parameters - "
925 LOCATE 5,1: PRINT "TestName [";TESTNAME$;"] :";
927 INPUT Z$: IF Z$ <> "" THEN TESTNAME$ = Z$
930 LOCATE 6,1: PRINT "Termination time for run [";TERMTIME%;"] (hr):";
932 INPUT Z$: IF Z$ <> "" THEN TERMTIME% = VAL(Z$)
935 LOCATE 7,1: PRINT "Termination voltage for run [";TERMVOLT;"] :";
937 INPUT Z$: IF Z$ <> "" THEN TERMVOLT = VAL(Z$)
940 LOCATE 8,1: PRINT "Maximum time between points [";MAXTIME3%;"] (min):";
942 INPUT Z$: IF Z$ <> "" THEN MAXTIME3% = VAL(Z$)
945 LOCATE 9,1: PRINT "Maximum voltage between points [";MAXVOLT3;"] :";
947 INPUT Z$: IF Z$ <> "" THEN MAXVOLT3 = VAL(Z$)
950 RETURN
999 '********** BATTERY CYCLING MODULE **********
1000 '
1005 '********** write parameter file **********
1010 OPEN "c:"+TESTNAME$+".HDR" FOR OUTPUT AS #1
1015 PRINT #1, TESTNAME$
1020 PRINT #1, DCUR; DCOV; CCUR; CCOV; MAXCYCLE%
1025 CLOSE #1
```

```
1030 ' ************** convert currents to GPIB string commands ***********
1035 IF DCUR > 0 THEN DCUR = – DCUR      'insure neg. I for discharge
1040 IF ABS(DCUR) < 2 THEN DCUR$=STR$(DCUR*1000)+ ",–6" ELSE IF
ABS(DCUR) > 20 THEN DCUR$=STR$(DCUR*10)+",–4" ELSE
DCUR$=STR$(DCUR*100)+ ",–5"
1045 IF CCUR < 0 THEN CCUR = – CCUR      'insure pos. I for charge
1050 IF ABS(CCUR) < 2 THEN CCUR$=STR$(CCUR*1000)+ ",–6" ELSE IF
ABS(CCUR) > 20 THEN CCUR$=STR$(CCUR*10)+",–4" ELSE
CCUR$=STR$(CCUR*100)+",–5"
1070 PROGRAM$= "BATTERY CONSTANT CURRENT CYCLING"
1075 GOSUB 9200
1080 LOCATE 3,1: PRINT "CURRENT CYCLE"
1085 LOCATE 3,40: PRINT "MAXIMUM CYCLES";MAXCYCLE%
1090 LOCATE 5,1: PRINT "CURRENT DATA POINT"
1095 LOCATE 6,7: PRINT "ELAPSED TIME"
1100 LOCATE 6,27: PRINT "min"
1105 LOCATE 7,12: PRINT "VOLTAGE"
1110 ' ***************** run experiment ************************************
1115 CYCLE% = 1
1120 IF FIRSTCYCLE$ = "D" THEN CUR$=DCUR$ ELSE CUR$=CCUR$
1125 PT% = 0
1130 LOCATE 3,15: PRINT USING "###";CYCLE%
1135 WRT$ = "SETI "+CUR$ : CALL IBWRT(PAR%,WRT$)      'set current on PAR
173
1140 CALL IBRD(HP%,RD$)      'start time & OCV
1145 ETIME(PT%) = 0 : EVOLT(PT%) = VAL(LEFT$(RD$,8)) * 10
1150 WRT$ = "CELL 1" : CALL IBWRT(PAR%,WRT$)    'cell on
1152 IF PT% = 2000 THEN GOSUB 1400          'array full
1155 PT% = PT% + 1 : STIME = TIMER
1160 OPEN TESTNAME$+".TMP" FOR APPEND AS #1
1165 PRINT #1,ETIME(PT%–1);EVOLT(PT%–1) : CLOSE #1
1170 LOCATE 5,20: PRINT USING "####"; PT%
1175 CTIME = TIMER : CALL IBRD(HP%,RD$)      'current time, voltage
1180 IF CTIME < STIME THEN GOTO 1315   'new day
1185 ETIME(PT%) = ETIME(PT%–1)+(CTIME – STIME)/60      'elapsed time in
minutes
1190 EVOLT(PT%) = VAL(LEFT$(RD$,8)) * 10
1195 LOCATE 6,20: PRINT USING "####.#";ETIME(PT%)
1200 LOCATE 7,20: PRINT USING "#.####";EVOLT(PT%)
1205 IF VAL(CUR$) < 0 AND EVOLT(PT%) <=DCOV THEN 1232   'end of discharge
1210 IF VAL(CUR$) > 0 AND EVOLT(PT%) >=CCOV THEN 1232   'end of charge
1215 IF ABS(ETIME(PT%) – ETIME(PT% – 1)) >= MAXTIME% THEN GOTO 1155
1220 IF ABS(EVOLT(PT%) – EVOLT(PT% – 1)) >= MAXVOLT THEN GOTO 1155
1225 GOTO 1175
1230 ' *************** end cycle ***************************************
1232 IF VAL(CUR$) < 0 AND FIRSTCYCLE$ = "D" THEN GOTO 1295
1234 IF VAL(CUR$) > 0 AND FIRSTCYCLE$ = "C" THEN GOTO 1295
1235 WRT$ = "CELL 0" : CALL IBWRT(PAR%,WRT$)   'cell off
1240 GOSUB 1335                ' print data
1245 IF CYCLE% = MAXCYCLE% THEN RETURN
```

```
1250 CYCLE% = CYCLE% + 1
1255 START = TIMER
1260 LOCATE 13,26:PRINT "min until next cycle begins";
1265 WHILE TIMER < START + DELAY%
1270 LOCATE 13,20 : PRINT USING "##.##";(START + DELAY% – TIMER)/60;
1275 WEND
1280 LOCATE 13,20: PRINT SPC(35);
1285 GOTO 1120
1290 ' *************** change to charge current ****************************
1295 IF FIRSTCYCLE$ = "D" THEN CUR$=CCUR$ ELSE CUR$=DCUR$
1300 WRT$ = "SETI "+CUR$ : CALL IBWRT(PAR%,WRT$)
1305 GOTO 1155                              'next point
1310 ' ***************** change in day and time adjusts ********************
1315 ETIME(PT%) = ETIME(PT%–1) + (86400! – STIME + CTIME)/60
1320 EVOLT(PT%) = VAL(LEFT$(RD$,8)) * 10
1325 GOTO 1155                          'force point at start of new day
1330 ' ****************** SAVE DATA FILES ***********************************
1335 OPEN TESTNAME$+".hdr" FOR APPEND AS #1
1340 PRINT #1, CYCLE%; EVOLT(0); PT%
1345 CLOSE #1
1350 OPEN TESTNAME$+".HIS" FOR APPEND AS #1   'open file for data
1355 FOR J% = 0 TO PT%
1360 PRINT #1, ETIME(J%); EVOLT(J%)
1365 NEXT J%
1370 CLOSE #1
1375 KILL TESTNAME$+".TMP"
1380 RETURN
1400 GOSUB 1350                          'add data to .HIS file
1410 PT% = –1                           'reintialized index
1420 RETURN
1999        '          **************          VOLTAGE–CURRENT          MODULE
*****************************
2000 PROGRAM$="BATTERY CURRENT – VOLTAGE DATA"
2005 GOSUB 9200
2025 ' ***************** run experiment ************************************
2030 START = TIMER
2035 FOR J% = 1 TO PT2%
2040 CUR(J%) = STARTI + (J%–1)*CURINCR                  'CALC CURRENT
2042 CUR$ = STR$(CUR(J%))
2044 IF INSTR(CUR$,"E") <> 0 THEN GOSUB 2200   'don't send exp not'ion to 276
2045 CUR = CUR(J%)
2050 IF CUR > 0 THEN CUR = – CUR
2055 IF ABS(CUR) < 2 THEN CUR$=STR$(CUR*1000)+ ",–6" ELSE IF ABS(CUR) >
20 THEN CUR$=STR$(CUR*10)+",–4" ELSE CUR$=STR$(CUR*100)+ ",–5"
2060 WRT$ = "SETI "+CUR$ : CALL IBWRT(PAR%,WRT$)       ' SET CURRENT
2065 WHILE TIMER < START + RECOVER% : WEND                ' RECOVERY
DELAY
2070 CALL IBRD(HP%,RD$): OCV(J%) = VAL(LEFT$(RD$,8))*10 ' READ OCV
2075 CALL IBWRT(PAR%,CELLON$) : START = TIMER             ' CELL ON
2080 WHILE TIMER < START + LOADTIME% : WEND          ' TIME ON LOAD
```

```
2085 CALL IBRD(HP%,RD$)                          ' READ LOAD VOLTAGE
2090 CALL IBWRT(PAR%,CELLOFF$) : START = TIMER          ' CELL OFF
2095 LOADV(J%) = VAL(LEFT$(RD$,8))*10              ' CALC LOAD VOLTAGE
2100 NEXT J%
2105 ' ****************** SAVE DATA FILES ***********************************
2110 OPEN TESTNAME$+".VI" FOR OUTPUT AS #1   'open file for data
2115 FOR J% = 1 TO PT2%
2120 PRINT #1, CUR(J%); OCV(J%);LOADV(J%);OCV(J%) - LOADV(J%)
2125 NEXT J%
2130 CLOSE #1
2135 RETURN
2199 ' ***************** TRUNCATE LONG STRING BEFORE SENDING TO PAR
276 ****
2200 SIGN$ = LEFT$(CUR$,1) : NUMBR$ = MID$(CUR$,2,1) + MID$(CUR$,4,2)
2220 P = INSTR(CUR$,"E") : PWR$ = MID$(CUR$,P+2,2)
2230 PSIGN$ = MID$(CUR$,P+1,1)
2240 IF PSIGN$ = "+" THEN CUR$ = SIGN$ + LEFT$(NUMBR$,VAL(PWR$)+1) +
"." + RIGHT$(NUMBR$,LEN(NUMBR$)-VAL(PWR$) + 1)
2250 CUR$ = SIGN$ + "." + STRING$(VAL(PWR$) - 1, "0") + NUMBR$
2260 CUR = VAL(CUR$)
2270 RETURN
2999 ' *************** OCV MONITOR MODULE ********************************
3000 PROGRAM$= "BATTERY OPEN CIRCUIT VOLTAGE"
3010 GOSUB 9200
3012 LOCATE 3,10: PRINT "Elapsed Time ";
3014 LOCATE 3,35: PRINT "Voltage ";
3020 PT%=0: CALL IBRD(HP%,RD$)
3025 ETIME(PT%)=0: EVOLT(PT%)=VAL(LEFT$(RD$,8))*10
3030 OPEN TESTNAME$+".OCV" FOR APPEND AS #1
3035 PRINT #1, ETIME(PT%);EVOLT(PT%): CLOSE #1
3040 PT%=PT%+1: STIME = TIMER
3045 CTIME = TIMER: CALL IBRD(HP%,RD$)
3050 IF CTIME < STIME THEN GOTO 3200
3055 ETIME(PT%) = ETIME(PT%-1) + (CTIME-STIME)/3600    'TIME IN HOURS
3060 EVOLT(PT%) = VAL(LEFT$(RD$,8))*10
3062 LOCATE 3,25: PRINT USING "##.######";ETIME(PT%);
3064 LOCATE 3,43: PRINT USING " #.####";EVOLT(PT%);
3065 IF ETIME(PT%) >= TERMTIME% THEN GOTO 3100
3070 IF EVOLT(PT%) <= TERMVOLT THEN GOTO 3100
3075 IF ABS(ETIME(PT%) - ETIME(PT% - 1))*60 >=MAXTIME3% THEN GOTO
3030
3080 IF ABS(EVOLT(PT%) - EVOLT(PT%-1)) >= MAXVOLT3 THEN GOTO 3030
3085 GOTO 3045
3100 OPEN TESTNAME$+".OCV" FOR APPEND AS #1
3105 PRINT #1, ETIME(PT%);EVOLT(PT%): CLOSE #1
3110 RETURN
3200 ETIME(PT%) = ETIME(PT%-1) + (86400! - STIME + CTIME)/3600
3210 EVOLT(PT%) = VAL(LEFT$(RD$,8))*10
3220 GOTO 3030
3999 ' *************** EXIT TO DOS ***************************************
```

```
4000 CLS: SYSTEM
4001 '
6999 ' ********** initialize devices on GPIB **********
7000 BDNAME$ = "PAR276": CALL IBFIND(BDNAME$,PAR%)
7010 IF PAR% < 0 THEN GOSUB 9000 : GOTO 7000
7020 BDNAME$ = "HP3455A": CALL IBFIND(BDNAME$,HP%)
7030 IF HP% < 0 THEN GOSUB 9000 : GOTO 7020
7040 CALL IBCLR(HP%): CALL IBCLR(PAR%)
7050 WRT$ = "F1 R3 T1 A1": CALL IBWRT(HP%,WRT$)
7090 RETURN
7999 '********** end experiment **********
8000 CLS
8010 CALL IBCLR(HP%): CALL IBCLR(PAR%)          'clear devices
8040 RETURN
8999 ' ********** device error subroutine **********
9000 CLS
9010 PRINT "ERROR – Device ";BDNAME$;" not found"
9020 PRINT "Check power and connections"
9030 PRINT "Press <RETURN> to try again or <ESC> to abort"
9040 Z$=INKEY$: IF Z$="" THEN 9040
9050 IF Z$ = CHR$(27) THEN GOSUB 8000: GOTO 110
9060 IF Z$ = CHR$(13) THEN RETURN
9070 BEEP: GOTO 9040
9099 ' ********** WARNING MESSAGE **********
9100 CLS: PRINT "Insure PAR 173 on EXT CEll and leads are connected to cell"
9110 PRINT:PRINT "Press <RETURN> to start experiment";
9120 Z$ = INKEY$: IF Z$ <> CHR$(13) THEN GOTO 9120
9130 CLS
9140 RETURN
9200 CLS:LOCATE 1,40 – LEN(PROGRAM$)/2 : PRINT PROGRAM$: RETURN
```

APPENDIX B
EXPLANATION OF EXPERIMENT SUBROUTINES

Battery Cycling Subroutine

Lines 1010–1025 saves the file (TESTNAME$.HDR) that records identifying information about the cycling experiment. Lines 1035–1050 convert the current values into string variables to be used as output to the potentiostat. Lines 1070–1105 display information on the screen while the subroutine is running. Line 1115 sets the cycle counter to 1, the first cycle while line 1120 selects the proper current for the first cycle. Line 1125 initializes the data point counter to zero. Line 1130 prints status information to the screen. Line 1135 sets the current on the postentiostat. Lines 1140–1145 read the open circuit voltage and stores the voltage at time zero in the data array. Line 1150 turns the cell on. Line 1155 increments the data pointer and starts the time for the current point. Lines 1160–1165 add the previously acquired point to the data file, and line 1170 displays the data on the screen. Line 1180 checks to see if the computer clock has passed midnight before evaluating the time in line 1185 and the voltage in line 1190. Lines 1195–1200 display the elapsed time and voltage on the screen. Lines 1205–1220 make various tests to determine if any of the limits has been reached and transfer control to the appropriate program line. Lines 1205 and 1210 check for end of discharge or charge, and transfer to line 1232 and 1234 which determine if the current needs to be changed or if the cycle is complete. If the condition in either lines 1232 or 1234 is true then program continues at line 1295 which changes the current appropriately. If both conditions are false then the program continues at line 1235 ending the current cycle.

If the current needs to be changed, line 1295 puts the appropriate current value in CUR$ and line 1300 sets the current on the potentiostat. Line 1305 then returns to line 1155 to take the next data point.

If a cycle has ended, line 1235 turns the cell off. Line 1240 goes to the routine that adds the data for the cycle to the permanent data files TESTNAME$.HDR and TESTNAME$.HIS. Line 1245 checks for completion of the last cycle. If the cycle just completed is the last cycle as determined by the parameters entered by the user, then the subroutine returns, otherwise the cycle counter is incremented in line 1250. Line 1255 starts timing for the delay between cycles and lines 1260 and 1270 print the delay time on screen. The WHILE – WEND loop in lines 1265–1775 delays for the next cycle. When the delay time ends, line 1280 clears the screen of the delay time and line 1285 sends control back to line 1120 to start the next cycle.

Lines 1215 and 1220 check for expiration of maximum time since last point and maximum voltage change since last point. If either event has occured, the program accepts the current point by shifting to line 1155. If none of the events in lines 1205–1220 have occured, line 1225 loops back to line 1175 to continue the process of monitoring time and voltage for the current data point.

Lines 1315–1235 evaluate the time and voltage when the elapsed time crosses midnight on the computer clock and returns to take the next data point.

A temporary data file (TESTNAME$.TMP) is saved in lines 1160 and 1165. This file maintains data on the current cycle so if power is lost or the program otherwise terminates, the data collected for this cycle is not lost. Lines 1335–1380 is a subroutine that adds data for the current cycle to a permanent file that already contains data from previous cycles and kills the temporary file.

Voltage–Current Subroutine

Lines 2000–2005 print the title on the screen. Line 2030 takes the starting time. Lines 2035–2100 are the data–taking loop for PT2% number of points. Line 2040 calculates the current for the current point and puts it in the CUR array. Line 2042 puts the current into string variable and Line 2044 checks for expoential form. If the number is in exponential form, the subroutine at Lines 2200 – 2270 converts the number to a form that the PAR 276 will except. Lines 2045–2055 put the current value into a string to be used in the command to the potentiostat and line 2060 then sets the current on the potentiostat. Line 2065 is a WHILE–WEND loop that delays the recovery time before reading the open circuit voltage in line 2070 and turning on the cell in line 2075. Line 2080 is a WHILE–WEND loop that determines the time the cell is on load. At the end of this loop, the voltage is read in line 2085 and line 2090 turns the cell off and restarts the timer for the next data point. Line 2095 stores the load voltage in the data array LOADV.

Once PT2% data points have been acquired, the data is saved in the file TESTNAME$.VI by lines 2110 to 2130 and the subroutine ends. Data saved in this file are the current, open circuit voltage, load voltage, and the difference between the load and open circuit voltages.

Open Circuit Voltage Monitor

Lines 3000 and 3010 print the title on the screen. Lines 3012, 3014, 3062, and 3064 print data on the screen as it is acquired. Line 3020 zeros the data point counter and reads the initial voltage. Line 3025 forces the initial data point for time zero and intial voltage. Lines 3030 and 3035 saves this first point to the data file TESTNAME$.OCV. Line 3040 increments the data counter and starts the timer. Line 3045 reads the current time and voltage. Line 3050 transfers to the routine that calculates the elapsed time if midnight has been passed on the computer clock. Line 3055 calculates the elapsed time in hours and line 3060 determines the voltage for the current point. Lines 3065 and 3070 test the time and voltage to see if either the maximum run time or termination voltage has been reached. If either is true then the current data is saved, and the subroutine returns. Lines 3075 and 3080 check to see if the required time has elasped or voltage has changed enough to take another data point. If either parameter is exceeded, then the program goes to line 3030 which saves the current point and the program continues with the next point. If neither parameter is met, then line 3085 sends control to line 3045 and a new voltage and time are determined. Lines 3200–3220 calculates the elapsed time when midnight has been passed.

APPENDIX C
UTILITY PROGRAMS LISTINGS

SEPARATE.BAS

```
10 KEY OFF
20 CLS
30 PRINT "Enter Testname ";
40 INPUT TESTNAME$
50 I=1: I$=RIGHT$(STR$(I),1)
100 OPEN TESTNAME$+".his" FOR INPUT AS #1
120 OPEN TESTNAME$+"."+I$ FOR OUTPUT AS #2
130 WHILE NOT EOF(1)
140 INPUT #1, ETIME,EVOLT
150 PRINT #2, ETIME,EVOLT
160 INPUT #1, ETIME,EVOLT
170 IF ETIME=0 THEN 300
180 PRINT #2, ETIME,EVOLT
190 GOTO 160
300 CLOSE #2
310 I=I+1: I$=RIGHT$(STR$(I),1)
320 OPEN TESTNAME$+"."+I$ FOR OUTPUT AS #2
330 PRINT #2, ETIME,EVOLT
340 GOTO 160
350 WEND
360 CLOSE #1,#2
370 END
```

VCONVERT.BAS

```
10 KEY OFF
20 CLS
30 PRINT "Enter Testname ";
40 INPUT TESTNAME$
50 PRINT "Did ";TESTNAME$;" start with [C]harge or [D]ischarge";
60 INPUT Z$: IF Z$ <> "" THEN FIRSTCYCLE$=Z$
70 IF INSTR("CcDd",FIRSTCYCLE$) = 0 THEN BEEP: GOTO 50
80 IF INSTR("Cc",FIRSTCYCLE$) = 0 THEN FIRSTCYCLE$="D" ELSE
FIRSTCYCLE$="C"
90 CHGTIME = 0
95 IF INSTR(TESTNAME$,".") <> 0 THEN GOTO 450 'HANDLE EXTENSION IN
NAME
97 EXT$ = "HIS"
100 OPEN TESTNAME$+".hdr" FOR INPUT AS #1
105 INPUT #1, FLNAME$
110 INPUT #1, DCUR,DCOV,CCUR.CCOV,MAXCYCLE%
120 CLOSE #1
130 IF DCUR < 0 THEN DCUR = -DCUR 'pos Q for discharge
```

```
140 IF CCUR > 0 THEN CCUR = -CCUR       'neg Q for charge
150 OPEN TESTNAME$+"."+EXT$ FOR INPUT AS #1
160 OPEN TESTNAME$+".dtv" FOR OUTPUT AS #2
165 IF FIRSTCYCLE$="C" THEN GOTO 245
170 IF EOF(1) THEN GOTO 310
180 INPUT #1, ETIME,EVOLT
185 IF FIRSTCYCLE$ = "D" THEN CTIME = ETIME - CHGTIME ELSE CTIME =
CHGTIME -(ETIME - CHGTIME)
200 PRINT #2, CTIME;EVOLT
210 IF EVOLT > DCOV THEN GOTO 170
220 IF FIRSTCYCLE$="C" THEN CHGTIME = 0 ELSE CHGTIME = ETIME
245 IF EOF(1) THEN 310
250 INPUT #1,  ETIME,EVOLT
260 IF FIRSTCYCLE$ = "C" THEN CTIME = ETIME - CHGTIME ELSE CTIME =
CHGTIME - (ETIME - CHGTIME)
280 PRINT #2, CTIME;EVOLT
290 IF EVOLT < CCOV THEN GOTO 245
300 IF FIRSTCYCLE$="D" THEN CHGTIME = 0 ELSE CHGTIME = ETIME
305 GOTO 170
310 CLOSE #1, #2
320 END
450 FLNAME$=MID$(TESTNAME$,1,INSTR(TESTNAME$,".")-1)
460 EXT$=MID$(TESTNAME$,INSTR(TESTNAME$,".")+1,LEN(TESTNAME$))
470 TESTNAME$=FLNAME$
480 GOTO 100
```

QCONVERT.BAS

```
10 KEY OFF
20 CLS
30 PRINT "Enter Testname ";
40 INPUT TESTNAME$
50 PRINT "Did ";TESTNAME$;" start with [C]harge or [D]ischarge";
60 INPUT Z$: IF Z$ <> "" THEN FIRSTCYCLE$=Z$
65 IF INSTR(TESTNAMEA$,".") <> 0 THEN GOTO 450
70 IF INSTR("CcDd",FIRSTCYCLE$) = 0 THEN BEEP: GOTO 50
80  IF  INSTR("Cc",FIRSTCYCLE$)  =  0  THEN  FIRSTCYCLE$="D"  ELSE
FIRSTCYCLE$="C"
90 COULOMBS = 0: CHGTIME = 0
95 IF INSTR(TESTNAME$,".") <> 0 THEN GOTO 450
97 EXT$="HIS"
100 OPEN TESTNAME$+".hdr" FOR INPUT AS #1
105 INPUT #1, FLNAME$
110 INPUT #1, DCUR,DCOV,CCUR,CCOV,MAXCYCLE%
120 CLOSE #1
130 IF DCUR < 0 THEN DCUR = -DCUR       'pos Q for discharge
140 IF CCUR > 0 THEN CCUR = -CCUR       'neg Q for charge
150 OPEN TESTNAME$+"."+EXT$ FOR INPUT AS #1
160 OPEN TESTNAME$+".DTQ" FOR OUTPUT AS #2
```

```
162 IF FIRSTCYCLE$="C" THEN QSIGN = -1 ELSE QSIGN = +1
165 IF FIRSTCYCLE$="C" THEN GOTO 245
170 IF EOF(1) THEN GOTO 310
175 IF FIRSTCYCLE$ = "D" THEN CHGTIME = 0: COULOMBS = 0
180 INPUT #1, ETIME,EVOLT
185 CTIME = ETIME - CHGTIME
190 CHARGE = COULOMBS + CTIME * DCUR/1000 * 60     'time to seconds & mA to
A
200 PRINT #2, CHARGE*QSIGN;EVOLT
210 IF EVOLT > DCOV THEN GOTO 170
220 GOSUB 400
230 IF FIRSTCYCLE$ = "C" THEN COULOMBS = 0: CHGTIME = 0
245 IF EOF(1) THEN 310
250 INPUT #1,  ETIME,EVOLT
260 CTIME = ETIME - CHGTIME
270 CHARGE = COULOMBS + CTIME * CCUR/1000 * 60
280 PRINT #2, CHARGE*QSIGN;EVOLT
290 IF EVOLT < CCOV THEN GOTO 245
300 GOSUB 400: GOTO 170
310 CLOSE #1, #2
320 END
400 COULOMBS = CHARGE
410 CHGTIME = ETIME
420 RETURN
450 FLNAME$=MID$(TESTNAME$,1,INSTR(TESTNAME$,".")-1)
460 EXT$=MID$(TESTNAME$,INSTR(TESTNAME$,".")+1,LEN(TESTNAME$))
470 TESTNAME$=FLNAME$
480 GOTO 100
```

TCONVERT.BAS

```
10 KEY OFF
20 CLS
30 PRINT "Enter Testname ";
40 INPUT TESTNAME$
95 IF INSTR(TESTNAME$,".") <> 0 THEN GOTO 450
97 EXT$="HIS"
150 OPEN TESTNAME$+"."+EXT$ FOR INPUT AS #1
160 OPEN TESTNAME$+".hrs" FOR OUTPUT AS #2
170 WHILE NOT EOF(1)
180 INPUT #1, ETIME,EVOLT
185 ETIME = ETIME / 60        'convert minutes to hours
200 PRINT #2, ETIME;EVOLT
300 WEND
310 CLOSE #1, #2
320 END
450 FLNAME$=MID$(TESTNAME$,1,INSTR(TESTNAME$,".")-1)
460 EXT$=MID$(TESTNAME$,INSTR(TESTNAME$,".")+1,LEN(TESTNAME$))
470 TESTNAME$=FLNAME$
480 GOTO 150
```